$2^{\text{ème}}$ année ENSI Caen
TP Optimisation: algorithme d'optimisation avec contrainte

Dans ce TP, nous allons nous intéresser à des problèmes d'optimisation avec contraintes, c'est-à-dire, de la forme:

$$(\Sigma) \qquad \begin{cases} \min f(x) \\ x \in C \end{cases}$$

où C est un ensemble non vide, fermé de $\mathbb{R}^n$. Nous considérerons plus particulièrement, des ensembles de contraintes de la forme:

$$C = \{x \in \mathbb{R}^n \,|\, g_i(x) = 0,\, i = 1, \ldots, p,\, h_j(x) \leq 0,\, j = 1, \ldots, m\}$$

où les fonctions g_i, $i = 1, \ldots, p$ et h_j, $j = 1, \ldots, m$ sont de classe C^1.

1 Méthode de Pénalisation

1.1 Principe de la méthode de pénalisation

Le principe de la méthode de pénalisation est de remplacer le problème avec contrainte

$$(\Sigma) \qquad \begin{cases} \min f(x) \\ x \in C \subset \mathbb{R}^n \end{cases}$$

par le problème suivant sans contrainte

$$(\Sigma_\epsilon) \qquad \begin{cases} \min \left(f(x) + \frac{1}{\epsilon} p(x)\right) \\ x \in \mathbb{R}^n \end{cases}$$

où $p : \mathbb{R}^n \to \mathbb{R}$ est une fonction de pénalisation des contraintes et $\epsilon > 0$. Pour que l'on puisse appliquer des algorithmes d'optimisation sans contrainte, nous utiliserons des fonctions avec de bonne propriétés mathématiques. Plus particulièrement, nous utiliserons des fonctions de pénalisation p où la pénalisation est dite extérieure car le problème est modifié seulement à l'extérieur de l'ensemble des contraintes.
Voici quelques exemples de fonctions de pénalisation pour différentes contraintes

- Contrainte $x \leq 0$: une fonction p est $p(x) = \|x^+\|^2$.
- Contrainte $g(x) = 0$: une fonction p est $p(x) = \|g(x)\|^2$.
- Contrainte $h(x) \leq 0$: une fonction p est $p(x) = \|h(x)^+\|^2$

où $\|.\|$ est la norme euclidienne de $\mathbb{R}^n$, $x^+ = (x_1^+, \ldots, x_n^+)$ et $x_i^+ = \max\{x_i, 0\}$.
Nous avons le résultat de convergence suivant:

Théorème 1 *Soit f une fonction continue et coercive. Soit C un ensemble fermé non vide. On suppose que p vérifie les propriétés suivantes:*

1. *p est continue sur $\mathbb{R}^n$.*
2. *$\forall x \in \mathbb{R}^n$, $p(x) \geq 0$*
3. *$p(x) = 0 \iff x \in C$*

On a alors:

- *$\forall \epsilon > 0$, (Σ_ϵ) a au moins une solution x_ϵ*
- *La famille $(x_\epsilon)_{\epsilon > 0}$ est bornée*
- *Toute sous-suite convergente extraite de $(x_\epsilon)_{\epsilon>0}$ converge vers une solution de (Σ) lorsque $\epsilon \to 0$.*

On obtient alors l'algorithme suivant de pénalisation extérieure:

Algorithme de la méthode de pénalisation

Initialisation: Choisir $x^{(0)} \in \mathbb{R}^n$, $\epsilon > 0$

Résoudre le sous problème (Σ_ϵ) $\begin{cases} \min \left(f(x) + \frac{1}{\epsilon} p(x)\right) \\ x \in \mathbb{R}^n \end{cases}$ avec $x^{(0)}$ le point d'initialisation.

1.2 Implémentation de la méthode de pénalisation

L'algorithme de pénalisation permet de changer le problème d'optimisation avec contrainte en un problème d'optimisation sans contrainte que l'on sait traiter. Nous allons utiliser ici l'algorithme du gradient avec la méthode d'Armijo pour le calcul du pas. Le but de cette section est de créer une fonction qui va appliquer l'algorithme de pénalisation à laquelle on donnera tous les paramètres nécessaires comme lors du premier TP, mais également le nom des fonctions f et p, afin d'avoir une fonction la plus réutilisable possible. Dans un premier temps, nous allons réécrire la fonction qui donne le pas en suivant la méthode d'Armijo et dans un second temps, nous réécrirons la fonction qui implémente la méthode du gradient.

1.2.1 Question 1

En se basant sur la fonction Armijo écrite lors du premier TP, écrire une fonction nommée "Armijo_penalisation" qui calcul le pas α en suivant la méthode d'Armijo pour la fonction $f(x) + \frac{1}{\epsilon}p(x)$, qui prend comme entrées:

- x_k: le point considéré
- d_k: la direction considérée
- fct_obj: une variable qui contient le nom de la fonction objectif
- $grad\_obj$: une variable qui contient le nom du gradient de la fonction objectif
- fct_p: une variable qui contient le nom de la fonction de pénalisation
- $grad\_p$: une variable qui contient le nom du gradient de la fonction de pénalisation
- *epsilon*: le paramètre ϵ.

et qui donne en sortie le paramètre α correspondant.
L'exemple suivant montre comment se servir du nom d'une fonction passée en paramètre.

Exemple 1 (Variable pour transmettre un nom de fonction)

- *Créer la fonction suivante:*

  function [y]=abcd(x)
  *y=2*x*

- *Définir la variable nom_fonction=@abcd*
- *Évaluer nom_fonction(1)*

1.2.2 Question 2

En se basant sur la fonction qui implémente l'algorithme du gradient écrite lors du premier TP, écrire une fonction nommée "algo_penalisation" qui prend en entrée:

- x_0: le point d'initialisation de l'algorithme,
- *epsilon*: la valeur du paramètre de pénalisation,
- fct_obj, fct_pena: variables qui contiennent les noms des fonctions qui décrivent la fonction objectif f et la fonction de pénalisation p,
- $grad\_obj, grad\_pena$: variables qui contiennent les noms des gradients de la fonctions objectif f et de la fonction de pénalisation p,
- nb_iter_max: le nombre maximal d'itérations,
- *tol*: la tolérance.

et qui donne en sortie:

- x: l'ensemble des points obtenus à chaque itération par l'algorithme
- nb_iter: le nombre d'itérations effectuées par l'algorithme.

1.2.3 Question 3

On considère le problème suivant:

$$(\Sigma_1) \qquad \begin{cases} \min\ f_1(x) \stackrel{\triangle}{=} 1 + x + \frac{1}{3}x^3 \\ x \geq 0,\ x \in \mathbb{R} \end{cases}$$

- Implémenter la fonction de pénalisation extérieure dite quadratique: $p(x) = \|x^-\|^2$ où x^- est défini par $x^- = \max\{-x, 0\}$ (On remarque que le terme de pénalisation $p(x)$ ne joue un rôle qu'à l'extérieur du domaine admissible $\mathbb{R}^+$)).
- Tracer la fonction objectif $f_1(x)$ et la fonction objectif pénalisée $f_1(x) + \frac{1}{\epsilon}p(x)$, pour $x \in [-10, 10]$ et $\epsilon = 1, 0.1, 0.01, 0.001$.
- Appliquer l'algorithme de pénalisation pour la résolution de ce problème, en prenant la fonction de pénalisation extérieure quadratique.

2 Méthode de dualité: Algorithme d'Uzawa

2.1 Principe de l'algorithme d'Uzawa

L'idée générale de l'algorithme d'Uzawa est d'utiliser le Langrangien $\mathcal{L}$ au lieu de la fonction f. Il y a plusieurs raisons pour lesquelles le Lagrangien est préféré à la fonction objectif:

- La fonction Langrangienne comprend à la fois la fonction f et les contraintes g et h et représente bien le problème.
- Nous avons vu qu'une condition nécessaire du premier ordre pour que x^* soit un minimum de f avec contraintes est que x^* soit un point critique de $\mathcal{L}$.

Le Langrangien du problème est donné par

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{i=1}^{p} \lambda_i g_i(x) + \sum_{j=1}^{m} \mu_j h_j(x)$$

L'algorithme d'Uzawa est basé sur la notion de point selle

Définition 1 *On appelle point-selle de $\mathcal{L}$ sur $\mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$ tout triplet $(x^*, \lambda^*, \mu^*) \in \mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$ vérifiant l'équation*

$$\mathcal{L}(x^*, \lambda, \mu) \leq \mathcal{L}(x^*, \lambda^*, \mu^*) \leq \mathcal{L}(x, \lambda^*, \mu^*), \qquad \forall (x, \lambda, \mu) \in \mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$$

On a alors le résultat suivant:

Théorème 2 *Supposons que f, g et h soient des fonctions de classe C^1 et que le triplet $(x^*, \lambda^*, \mu^*) \in \mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$ soit un point-selle de $\mathcal{L}$ sur $\mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$. Alors, ce triplet vérifie les conditions KKT.*
Si de plus les fonctions f, g et h sont convexes, alors le triplet (x^*, λ^*, μ^*) est un point selle de $\mathcal{L}$ sur $\mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$ si et seulement si il vérifie les conditions KKT.

Le théorème précédent nous dit que nous devons chercher un point selle du Lagrangien. Nous allons chercher ce point selle de la façon suivante:

1. pour (λ^*, μ^*) fixés dans $\mathbb{R}^n \times (\mathbb{R}^+)^m$, nous allons chercher le minimum sans contraintes (sur $\mathbb{R}^n$) de la fonction $x \to \mathcal{L}(x, \lambda^*, \mu^*)$.
2. pour x^* fixé dans $\mathbb{R}^n$, on cherche le maximum sur $\mathbb{R}^p \times (\mathbb{R}^+)^m$ (c'est-à-dire des contraintes de bornes simples) de la fonction $(\lambda, \mu) \to \mathcal{L}(x^*, \lambda, \mu)$.

On effectue ces deux calculs consécutivement, et on obtient l'algorithme d'Uzawa.

Algorithme d'Uzawa

Initialisation: $l = 0$, choisir $\lambda^{(0)} \in \mathbb{R}^p$ et $\mu^{(0)} \in \mathbb{R}^m$
erreur=1
Boucle: Tant que $erreur > tol$:

a) calculer $x^{(l+1)} \in \mathbb{R}^n$ solution de
$$\left(\Sigma^{(l)}\right) \begin{cases} \min \mathcal{L}(x, \lambda^{(l)}, \mu^{(l)}) \\ x \in \mathbb{R}^n \end{cases}$$

b) calculer $\lambda^{(l+1)}$ et $\mu^{(l+1)}$ avec
$$\begin{cases} \lambda_i^{(l+1)} = \lambda_i^{(l)} + \rho g_i(x^{(l)}),\ i = 1, \dots, p \\ \mu_j^{(l+1)} = \max\left(0, \mu_j^{(l)} + \rho h_j(x^{(l)})\right),\ j = 1, \dots, m \end{cases}$$
ou $\rho > 0$ est un réel fixé (par l'utilisateur)

c) $erreur = \|x^{(l+1)} - x^{(l)}\|$

On a le résultat de convergence suivant.

Théorème 3 *On suppose que f est C^1 et elliptique, que g est affine, h convexe de classe C^1 et que g et h sont lipschitziennes.*
On suppose de plus que le Lagrangien $\mathcal{L}$ possède un point selle (x^*, λ^*, μ^*) sur $\mathbb{R}^n \times \mathbb{R}^p \times (\mathbb{R}^+)^m$.
Alors il existe ρ_1, ρ_2, avec $0 < \rho_1 < \rho_2$ tels que $\forall \rho \in [\rho_1, \rho_2]$, la suite $(x^k)_{k>0}$ généré par l'algorithme d'Uzawa converge vers x^*. De plus, ρ_2 est donnée par:

$$\rho_2 = \frac{\alpha}{2(M_g^2 + M_h^2)}$$

avec α la constante d'ellipticité de f, M_g et M_k, les constantes de Lipschitz associées à g et h.

2.2 Implémentation de l'algorithme d'Uzawa

Comme pour la méthode pénalisation, le but ici est de se ramener à un problème d'optimisation sans contrainte. On peut voir dans l'algorithme d'Uzawa qu'il faut trouver un minimum. On utilisera donc l'algorithme du gradient avec la méthode d'Armijo.

2.2.1 Question 4

En se basant sur la fonction Armijo écrite lors du premier TP, écrire une fonction "Armijo_Uzawa" qui calcul le pas α en suivant la méthode d'Armijo pour la fonction $f(x) + g(x)\lambda + h(x)\mu$, qui prend comme entrées:

- x_k: le point considéré
- d_k: la direction considérée
- fct_obj, fct_g, fct_h: contiennent les noms de la fonction f, la fonction g et la fonction h respectivement.
- $grad\_obj, grad\_g, grad\_h$: contiennent les noms des gradient de la fonction objectif, la fonction g et la fonction h respectivement.
- μ_l, λ_l: les paramètre μ et λ à l'instant l

et qui donne en sortie le paramètre α correspondant.

2.2.2 Question 5

En se basant sur la fonction qui implémente l'algorithme du gradient écrite lors du premier TP, écrire une fonction nommée "algo_Uzawa" qui prend en entrée: x_0, λ_0, μ_0, ρ, fct_obj, fct_g, fct_h, $grad\_obj$, $jacob\_g$, $jacob\_h$, nb_iter_max, tol, et qui donne en sortie: x, nb_iter.
Les conventions suivantes devront être respectées:

- Les fonctions fct_g et fct_h devront rendre en sortie, un vecteur colonne de dimension p et m respectivement.
- Les fonctions $jacob\_g$ et $jacob\_h$ rendront des vecteurs de dimension $p \times n$ et $m \times n$ en sortie, respectivement.
- Dans le cas où il n'y a pas de contrainte d'égalité (d'inégalité, resp.), la fonction fct_g (fct_h, resp.) rendra la valeur 0 en sortie, et la fonction $jacob\_g$ ($jacob\_h$) rendra un vecteur ligne nul de dimension n, de plus, l'algorithme d'optimisation sera initialisé avec $\lambda_0 = 0$ ($\mu_0 = 0$, resp.)

2.2.3 Question 6

On considère le problème suivant:

$$(\Sigma_2) \qquad \begin{cases} \min\ f_2(x,y) \triangleq 2x^2 + 3xy + 2y^2 \\ x \leq -\frac{1}{2}, \quad y \leq -\frac{1}{2}, \quad (x,y) \in \mathbb{R}^2 \end{cases}$$

- Résoudre le problème d'optimisation (Σ_2) avec l'algorithme d'Uzawa.
- Déterminer une fonction de pénalisation p pour le problème (Σ_2).
- Résoudre le problème d'optimisation (Σ_2) à l'aide de la méthode de pénalisation.
- Peut on résoudre le problème d'optimisation (Σ_1) avec l'algorithme d'Uzawa?

2.2.4 Question 7

On considère le problème suivant:

$$(\Sigma_3) \qquad \begin{cases} \min f_3(x) \triangleq x^2 + 10\sin(x) \\ x \in [2, 10] \end{cases}$$

- Tracer la fonction f_3 pour $x \in [-2, 16]$.
- Résoudre le problème d'optimisation (Σ_3) avec les algorithmes de pénalisation et d'Uzawa.

2.2.5 Question 8

On considère le problème suivant:

$$(\Sigma_4) \qquad \begin{cases} \min f_4(x) \triangleq x^T A x, \quad A = \begin{bmatrix} 5 & 3 & 1 \\ 3 & 2 & 1 \\ 1 & 1 & 4 \end{bmatrix} \\ x_1 + x_2 \leq x_3, \quad 3x_1 - 2x_2 = x_3 \end{cases}$$

- Quelle est la solution du problème d'optimisation (Σ_4)
- Résoudre le problème d'optimisation (Σ_4) avec les algorithmes de pénalisation et d'Uzawa.

2.2.6 Question 9 - distance d'un point à un hyperplan

On cherche à déterminer numériquement la distance d'un point $x_0 \in \mathbb{R}^n$ à un hyperplan $\mathcal{H}$ d'équation $Ax = b$, où A est un vecteur ligne de taille n et b est un réel. On peut écrire ce problème comme un problème d'optimisation sous contrainte:

$$(\Sigma_5) \qquad \begin{cases} \min f_5(x) \triangleq \frac{1}{2}(x - x_0)^T (x - x_0) \\ Ax = b \end{cases}$$

- Résoudre le problème d'optimisation (Σ_5) avec les algorithmes de pénalisation et d'Uzawa, pour les valeurs suivantes: $n = 4$, $x_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^T$, $A = \begin{bmatrix} 3 & 1 & 0 & 2 \end{bmatrix}$ et $b = 1$.
- Vérifier les résultats obtenus avec la solution exacte:

$$x^* = x_0 + A^T (A A^T)^{-1} (b - A x_0)$$

2.2.7 Question 10 - Optimisation d'un portefeuille d'actions

Supposons que l'on possède n actions, que l'on représente par des variables aléatoires $R_1, \ldots, R_n$. Chaque action rapporte en moyenne à l'actionnaire $e_i = \mathbb{E}(R_i)$ (espérance de R_i) au bout d'un an. On suppose que l'on investit une somme S donnée, et l'on note $x_i \in \mathbb{R}^+$ la proportion de la somme investie dans l'action i. Ainsi, on a:

$$\sum_{i=1}^{n} x_i = 1$$

Le portefeuille total est représenté par la variable aléatoire $R = \sum_{i=1}^{n} x_i R_i$ et rapporte donc en moyenne $\mathbb{E}(R) = \sum_{i=1}^{n} x_i e_i$. On désire imposer un rendement donné $r_0 > 0$, ce qui se traduit par:

$$r_0 = \sum_{i=1}^{n} x_i e_i$$

On modélise le risque du portefeuille d'actions par

$$\sigma^2(x) = \mathbb{E}[(R - \mathbb{E}(R))^2]$$

On note $A = (a_{ij})_{1 \leq i,j \leq n}$ la matrice de covariance définie par la relation $a_{ij} = \mathbb{E}[(R_i - \mathbb{E}(R_i))(R_j - \mathbb{E}(R_j))]$, $i, j = 1, \ldots, n$. On peut alors écrire que $\sigma^2(x) = \langle x, Ax \rangle$. On appelle f_6 la fonctionnelle définie sur $\mathbb{R}^n$ par:

$$f_6(x) = \frac{1}{2}\langle x, Ax \rangle$$

On appelle également K, l'ensemble des contraintes:

$$K \stackrel{\triangle}{=} \{x \in \mathbb{R}^n \mid \langle x, u \rangle = 1 \text{ et } \langle x, e \rangle = r_0\}$$

Le but de cette section est de déterminer numériquement la solution du problème:

$$(\Sigma_6) \qquad \begin{cases} \min\ f_6(x) \\ x \in K \end{cases}$$

- Application numérique: supposons $n = 5$, que $e_i = i$, pour $i = 1, \ldots, 5$ et $r_0 = 2.5$. On va générer la matrice A de la façon suivante:

```
A=diag(e./n);
R=rand(n,n);
A=A+0.1.*R'*R;
```

- Appliquer les algorithme de pénalisation et d'Uzawa pour la résolution du problème d'optimisation (Σ_6).
- Quel inconvénient constatez vous ici?
- Rajouter la contrainte $x \geq 0$ et appliquer de nouveau les algorithmes de pénalisation et d'Uzawa.